Journal of the Audio Engineering Society
Vol. 65, No. 1/2, January 2017 (© 2017)
DOI: https://doi.org/10.17743/jaes.2016.0069

# A Machine-Learning Approach to Application of Intelligent Artificial Reverberation

**EMMANOUIL T. CHOURDAKIS AND JOSHUA D. REISS,** *AES Member*
(e.t.chourdakis@qmul.ac.uk) (joshua.reiss@qmul.ac.uk)

*Queen Mary University of London, Mile End Road, London E14NS, UK*

We propose a design of an adaptive digital audio effect for artificial reverberation, controlled directly by desired reverberation characteristics, that allows it to learn from the user in a supervised way. The user provides monophonic examples of desired reverberation characteristics for individual tracks taken from the Open Multitrack Testbed. We use this data to train a set of models to automatically apply reverberation to similar tracks. We evaluate those models using classifier f1-scores, mean squared errors, and multi-stimulus listening tests.

## 1 INTRODUCTION

Digital Audio Effects (DAFx) are transformations on an audio signal, or a set of audio signals, where the transformation depends on a set of control parameters. In general, users of DAFx control these parameters themselves and they tend to change these parameters over time based on how the audio sounds. They assign specific audio features (or their changes) to specific parameters (or changes). Unknowingly, they are doing a form of classification where the samples are features of the audio and the classes are parameter sets. Our goal is to simulate this process using a supervised learning approach to train classifiers so that they automatically assign effect parameter sets to audio features. This way, we can train our reverberation effect to decide how to choose its parameters based just on the observed audio.

In order to create a reverberation effect that applies reverb automatically, we need to train it. Training can be done a-priori by, e.g., an expert user of the reverberation effect or on-line by the user of such an effect. Training is a process that involves user-interaction with the effect and so the parameters to be trained must make sense to the user. In [1] the authors provide a mapping from the delays and gains of a Moorer reverberator [2] to such parameters and thus make such an architecture suitable for our work.

Audio sources can be characterized by a multitude of features. Musical instrument tracks, for example, can be characterized by timbre, tempo, etc. An automatic reverberator trained on a set of audio is expected to be able to apply reverberation correctly on similar audio. For this reason, in order to create a reverberation effect that is as general as possible, we need to train it to a large and diverse set of audio data.

In order to train our system we perform feature selection to select the best features from a 31-dimensional feature space from 8 features found in the literature. Smoothing is then applied on the resulting features. We then compare 4 different classifiers on the classification task where our samples are vectors of audio features and classes are the parameter-set clusters. The training data consists of the control parameters provided by the user with a simple interface that allows her to control a simple reverberation effect. Testing is performed using cross-validation and multi-stimulus MUSHRA-style [3] tests.

An initial approach to intelligent artificial reverberation appeared in [4]. This has been extended in this paper by including mapping from characteristics of the desired impulse response, as given in [1], and by evaluating the resulting models using listening tests.

## 2 PREVIOUS WORK

There has been a lot of research in adaptive digital audio effects for automatic multitrack mixing but in almost all cases they focus on achieving a pre-specified goal. Parameter automation and intelligent control have been applied to many of the most popular audio effects (e.g., gain and faders [5], equalization [6], panning [7], and dynamic range compression [8]), but to the best of the authors' knowledge it has not been attempted on artificial reverberation. Furthermore, all of the above mentioned approaches except [5], which uses linear dynamical systems to estimate mixing weight coefficients, use fixed rules rather than arbitrary rules that are learned from training data. On the other hand, to the knowledge of the authors, there are no published works on automatic application of reverberation.

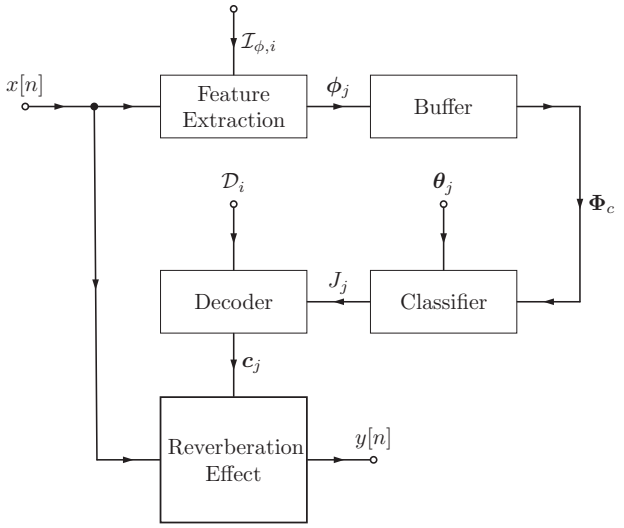


Fig. 1. Reverb application.

Table 1. Algorithmic Reverberation Parameters.

| Parameter (unit) | Controls | Min | Max |
|---|---|---|---|
| $d_1(s)$ | Comb filter array | 0.010 | 0.900 |
| $d_a(s)$ | All-pass filter array | 0.006 | 0.012 |
| $g_1$ | Comb filter array | 0.136 | 0.999 |
| $g_c$ | Low-pass filter array | 0.001 | 0.999 |
| $G$ | Dry/Wet mix | 0.001 | 0.999 |

Table 2. Perceptual Impulse Response Parameters. *SR* is the sampling rate.

| Parameter (Unit) | Controls | Min | Max |
|---|---|---|---|
| $T_{60}$ (s) | 60 *dB*-Reverberation Time | 0.02 | 4 |
| $D$ (echoes/s) | Echo Density | 1000 | 10000 |
| $C$ (dB) | Clarity | $-20$ | 10 |
| $T_c$ (s) | Central Time | 0.01 | 2 |
| $S_C$ (Hz) | Spectral Centroid | 200 | *SR*/4 |

Key work for the current paper can be found in [1] where they present the mapping from the reverberation parameters to measurements of the reverberation. In that paper the authors do not go as far as to provide a mapping from the measurements to the parameters, but they allow the control of the reverberation effect using high level descriptive terms. A similar work can also be found in [9] where the authors present a real-time feedback delay network (FDN) reverberator that allows control of perceptually relevant descriptors. Work using semantic descriptors can be found in [10] where they use a reverberation effect among others for their Semantic Audio Feature Extraction (SAFE) project, which allows users to assign high level descriptive terms to low level audio feature changes that are caused by effect parameter changes. In a similar fashion, [11] created a map of high level descriptive terms that correspond to low level reverberation effect parameters. Relevant work in [12] performs classification for drum sounds in order to control effect parameters but still relies on fixed rules.

## 3 EFFECT ARCHITECTURE

Our proposed design uses the traditional adaptive DAFx design [13] limited to one track and can be seen in detail in Fig. 1. It consists of an algorithmic reverberation effect where the values of the parameters are decided by a classifier model. The classifier model can be trained on-line or off-line. The architecture of the model training process can be seen in Fig. 2 where $\boldsymbol{\phi}_i$ is the feature vector of the $i$-th frame, $\boldsymbol{\Phi}_i$ is a matrix of features that consists of the vertical concatenation of the feature vectors (as row vectors), from frame 1 to frame $i$. In a similar fashion, $\boldsymbol{p}_i$ is the vector of desired characteristics of the impulse response provided by the user, $\boldsymbol{c}_i$ is the low level filters parameter vector to which $\boldsymbol{p}_i$ are mapped. $\boldsymbol{\theta}_i$ is the classifier parameter vector returned as result of the training after the $i$-th frame, and $\boldsymbol{\mathcal{D}}_i$ a dictionary that maps class labels to reverberator parameter sets.

Note that several features require the accumulation of a number of samples in a buffer (i.e., spectral features) to be computed. In such cases, latency equal to size of the buffer $\times$ the size of the frame is introduced. Similarly, some classifier models require the accumulation of several values before being able to make a decision and therefore introduce latency equal to a number of previous values $\times$ buffer size $\times$ size of each frame. Consequently, our architecture, although implementable in real time, can introduce latency that depends on the features chosen, as well as the models used. Therefore one should be careful in their choice of features and classifier model.

For our reverberation effect we use the algorithmic design given in [1] (Fig. 3) because of its simplicity and the fact that it can be trained directly from measurements of the reverberation. While this design has stereo input and output, we use it with monophonic signals that are split into stereo. We retain the stereo output in order to have a more natural sounding reverberation effect.

Our architecture is not limited to just this particular reverberation model. Any algorithmic reverberation model may be used as long as we can derive similar mappings between characteristics of the reverberation impulse response and low level filter parameters. The parameters of the reverberation effect and their limits can be seen in Table 1.

Those parameters are directly mapped to characteristics of the reverberation impulse response (Table 2). As we see in Fig. 3, we have an array of comb filters in series with two cascades comprised of an all-pass filter, a low-pass filter, and a gain. The delay times are linearly distributed with a ratio 1 : 1.5 with $d_1$ being the largest delay, and $d_6$ the smallest. The gains are chosen in order to keep the reverberation time the same for all comb filters (for more details on these parameters, please see [14]):

$$d_k = (1.5)^{1-k} d_1 \tag{1}$$

$$g_k = g_1^{1.5^{1-k}} \tag{2}$$

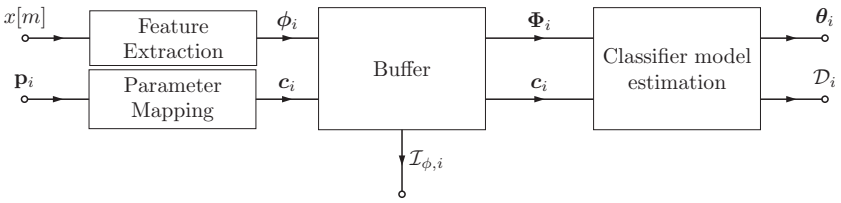


Fig. 2. Training of classifier models.

Distributing the comb filter delays with a constant ratio will result in overlapping echoes. For this problem, we use the solution given in [1], which is to make sure the delays are rounded to a co-prime set of integers when converted to samples. The delay times of both all-pass filters are controlled by a single parameter $d_a$ with a small delay difference $m$ (arbitrarily chosen, 2 *ms* in our case) between left and right channel. All-pass gains are set to 0.707. Finally, $g_c$ and $G$ control the low-pass filter and the gain, respectively.

The mappings from the reverberation effect parameters (Table 1) to reverberation characteristics (Table 2) can be written as [1]:

$$T_{60} = \frac{d_1}{\log g_1} \log\left(\frac{0.001\sqrt{2}}{(1-g_c)G}\right) \tag{3}$$

$$D = \frac{0.1}{d_a d_1} \sum_{k=1}^{6} 1.5^{k-1} \tag{4}$$

$$C = -10\log_{10}\left(\frac{1-g_c}{1+g_c} G^2 \sum_{k=1}^{6} \frac{g_1^{2\cdot 1.5^{1-k}}}{1-g_1^{2\cdot 1.5^{1-k}}}\right) \tag{5}$$

$$T_c = d_1 \frac{\sum_{k=1}^{6} \frac{1.5^{1-k} g_1^{2\cdot 1.5^{1-k}}}{(1-g_1^{2\cdot 1.5^{1-k}})^2}}{\sum_{k=1}^{6} \frac{1.5^{1-k} g_1^{2\cdot 1.5^{1-k}}}{1-g_1^{2\cdot 1.5^{1-k}}}} + d_a \tag{6}$$

$$S_c = \frac{\sum_{n=0}^{F_s/2} \frac{n}{1+g_c^2-2g_c\cos(2\pi n/F_s)}}{\sum_{n=0}^{F_s/2} \frac{1}{1+g_c^2-2g_c\cos(2\pi n/F_s)}} \tag{7}$$

From the last equation we can numerically compute $g_c$ (e.g., using Newton's method). The rest give us a (non-convex) 4 × 4 system. Given a set of target IR characteristics $(T_{60}^+, D^+, C^+, T_c^+, S_c^+)$ we approximate a set $(d_1', d_a', g_1', g_c', G')$ that brings the actual characteristics close to these values (see Appendix A.2). The careful reader will notice that there is no apparent relation between the minimum and maximum values given in Table 1 and those given in Table 2. For the solution of our system, these values were considered independent. Providing a definite relationship between reverberation parameters and IR characteristics proved very difficult due to the non-linear nature of the problem.

## 4 FEATURE EXTRACTION

Application of reverberation to a track can depend on the instrumentation, the type of music, and the percussiveness of the track, among others. For our task we use eight different features. Their names and their role can be seen in Table 3 [15–17]. The reason for choosing these features is because they have been used extensively in the literature for classification of instruments based on the above characteristics.

Before extracting the features from our audio, we first split the audio into 23 ms frames (1024 samples at 44.1 Hz) using the onset-based audio segmentation method described in [18], which is based on the *Spectral Contrast* feature. The reason for choosing this kind of segmentation, as shown in the original paper, is that it appears to give higher classification accuracies for at least the

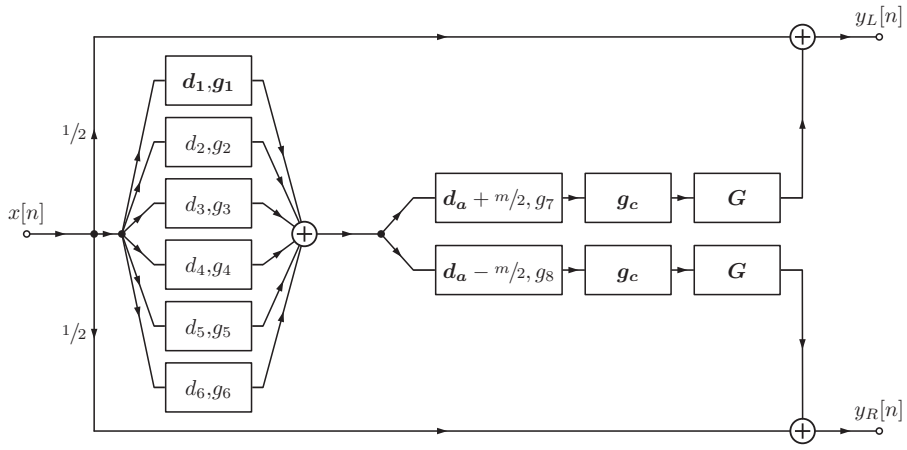


Fig. 3. The Moorer Reverberator used [1, 2]. Filter boxes are represented by their control parameters. $d_\nu$, $g_\nu$ for $\nu = 1 \ldots 6$ represent Comb Filter delays and gains respectively, $d_a$ all-pass filter delays, $g_c$ low-pass filter gain, and $G$ is a dry/wet mixer gain. Characteristics of the reverberation are mapped to the parameters shown in bold.

Table 3. Used features and their usage in the literature.

| Feature | Used in |
|---|---|
| *ZeroCrossingRate* | Instrument Identification<br>Source Identification |
| 13 *MFCCs* | Instrument Identification<br>Genre Classification |
| 12 *Spectral Contrast Coefficients* | Instrument Identification<br>Genre Classification |
| *Root Mean Square* | Instrument Identification<br>Voice/Music Discrimination<br>Audio Activity Detection |
| *Crest Factor* | Instrument Identification |
| *Spectral Centroid* | Instrument Identification<br>Genre Classification |
| *Spectral Roll-off* | Instrument Identification<br>Genre Classification |
| *Spectral Flux* | Instrument Identification |

music-genre classification task. We then concatenate our features into a 31 dimensional vector[1] for each frame. Next, we use Principal Component Analysis to filter out nonseparable or noisy features and reduce our feature vectors' dimensionality [19].

## 5 CLASSIFICATION AND TRAINING

We use classification on the audio features in order to control the values of the reverberation effect parameters. Given short excerpts of audio tracks together with the desired reverb characteristics, for training (Fig. 2):

(1) Convert the given reverberation characteristics $\boldsymbol{p}_i$ of values $[T_{60,\,i},\ D_i,\ C_i,\ T_{c,i},\ S_{C,i}]^T$ to a set of filter parameters $\boldsymbol{c}_i = \left[d_{1,i}\ d_{a,i}\ g_{1,i}\ g_i\ G_i\right]^T$ and add $\boldsymbol{c}_i$ to a set $\mathcal{C}$. $\mathcal{C}$ is the set of *parameter classes* and its cardinality $|\mathcal{C}|$ is the number of parameter classes. $\mathcal{C}_m$ denotes the $m$-th element of $\mathcal{C}$.

(2) Assign a label $J_i$ to the $i$-th frame if the chosen parameters for that frame belong to a class in $\mathcal{C}$:

$$J_i = \sum_{k=1}^{|\mathcal{C}|} \left(k \cdot \delta\left[\|\mathcal{C}_k - \mathbf{c}_i\|\right]\right) \tag{8}$$

$\|\cdot\|$ denotes a vector norm and $\delta[\,\cdot\,]$ is Kronecker's delta. Each number $J_i$ is the class label for the $i$-th frame. Keep a dictionary structure $\mathcal{D}_i$ for the classes introduced up to that point, comprised of the parameter sets and the labels to which they correspond:

$$\mathcal{D}_i = \{(J_k, \mathcal{C}_k) : k = 1, \ldots, |\mathcal{C}|\} \tag{9}$$

(3) Segment the audio excerpts into frames and calculate a 31-dimensional feature vector $\boldsymbol{\phi}_i$ for each frame:

$$\boldsymbol{\phi}_i = \left[\phi_{1,i}\ \phi_{2,i}\ \ldots\ \phi_{31,i}\right]^T \tag{10}$$

(4) The vectors $\boldsymbol{\phi}_i^T$ are vertically concatenated to form a matrix $\tilde{\boldsymbol{\Phi}}_i$ which is smoothed across columns with a Gaussian window (as a column vector of 41 elements). We then perform principal feature analysis [19] on the resulting matrix to derive $\boldsymbol{\Phi}_i$. We save the selected column numbers as the set $\mathcal{I}_{\phi,i}$.

(5) Use matrix $\boldsymbol{\Phi}_i$ together with the labels $J_i$ to estimate the parameters $\boldsymbol{\theta}_i$ of the chosen classifier.

From the training stage above we store the dictionary $\mathcal{D}_i$ and the classifier parameters $\boldsymbol{\theta}_i$. For the application of reverberation:

(1) Segment the audio track, to which we want to apply reverb, into frames and calculate a feature vector $\boldsymbol{\phi}_j$ for each frame. For each $\boldsymbol{\phi}_j$, we keep the rows the numbers of which are in $\mathcal{I}_{\phi,i}$.

(2) Use the classifier to select a label $J_j$ for the $j$-th frame.

(3) Use the dictionary structure $\mathcal{D}_i$ derived in the training phase as a function in order to convert from class labels $J_j$ to reverberation effect parameters $\boldsymbol{c}_j$ for each frame:

$$\boldsymbol{c}_j = \mathcal{D}_i[j] \tag{11}$$

where $\mathcal{D}_i[j] = \mathcal{C}_{J_j}$.

We compare four different classifiers: Gaussian Naive Bayes Classifier [20], One-vs-All Linear Support Vector Machine (SVM) Classifier [21], Hidden Markov Model Maximum A-posteriori Classifier [20], and a hybrid HMM classifier with observations taken from a set of SVMs [22]. Each classifier can be completely described by a vector of parameters $\boldsymbol{\theta}$. Given a set of training data, each of these classifiers are trained in a different way (usually using a variation of the EXPECTATION-MAXIMIZATION algorithm) to estimate their parameters.

In order to train our classifier models, we use excerpts from 254 audio files taken from the Open Multitrack Testbed [23]. First, the audio data is segmented into meaningful parts (i.e., song phrases, guitar solo parts, etc.) using a similarity matrix and a novelty function as found in [17]. A user is presented with a simple GUI where she can listen and apply reverb to the extracted parts by choosing the characteristics of the reverberation seen in Table 2. For the purpose of conducting listening tests, we asked 3 people familiar with the effect of reverberation to train our models. The segmented parts are split into frames using the method described in [18] and a tuple of features and parameters are extracted for each frame as described in Sec. 4. Features are filtered with a low-pass filter. The resulting data set is used to train the models described in Sec. 5. All our models were implemented in the PYTHON programming language using the SCIPY [24] library for Machine Learning and the ESSENTIA library [25] for onset segmentation, feature extraction, and storage[2].

[1] *MFCCs* and *Spectral Contrast* features have 13 and 12 dimensions respectively.

[2] Supplementary material for this research can be found at https://code.soundsoftware.ac.uk/projects/chourdakisreiss2016

Table 4. Average weighted f1-scores. Highest scores for each set are in bold. $|\mathcal{C}|$ is the number of classes calculated for each set.

| Training Set | $\|\mathcal{C}\|$ | GNB | SVM | HMM | $\text{HMM}^{\text{SVM}}$ |
|---|---|---|---|---|---|
| 1 | 7 | 0.79 | **0.82** | 0.70 | 0.70 |
| 2 | 9 | 0.80 | **0.81** | 0.69 | 0.49 |
| 3 | 6 | **0.81** | 0.79 | 0.75 | 0.73 |
| 4 | 8 | **0.78** | 0.77 | 0.65 | 0.59 |
| 5 | 7 | **0.82** | **0.82** | 0.73 | 0.60 |
| 6 | 7 | **0.87** | **0.87** | 0.73 | 0.52 |

## 6 RESULTS

We tested our models both by measuring classification performance as well as conducting a multi-stimulus MUSHRA-style [3] listening test.

### 6.1 Classification Performance

In order to validate our models, we split our data into 6 sets in order to reduce training times. Every set included 45 audio files except the last which included 29. Every file was a part of a Bass, Keyboards, Vocals, Percussion or Saxophone track. The files were randomly split into sets. In order to validate our classification scheme, we define the weighted macro f1-score for K classes:

$$f_1 = \sum_{k=1}^{K} \frac{n_k}{n} \cdot \frac{2tp_k}{tp_k + fp_k + fp_k} \tag{12}$$

In the definition above: $n_k$ is the number of samples belonging to class $k$, $n$ the total number of samples, $tp$ the number of samples classified correctly as belonging to class $k$, $fp$ the number of samples classified incorrectly as belonging to class $k$, and $fn$ the number of samples that belong to class $k$ but classified incorrectly to some other class. We validate our models as such:

(1) Split every set into 10 parts.
(2) Use 9 parts for training and 1 for testing. Do this for every combination of 10 parts. Store the predicted labels as well as the metrics $tp$, $fp$, and $fn$ for every run.
(3) Measure the weighted macro f1-scores for the predicted values.

We also use cross-validation to estimate the most suitable Markov chain number for our sequential models, as well as the number of Gaussian components for the case of the HMM with Gaussian emission distribution. Using the full training set we can see the overall *weighed f1-scores* in Table 4 and the average *Mean Squared Errors* in Table 5.

The high f1-scores are important because they represent the rate of agreement, between the automatic reverberation effect and the user that trained it, on the parameters of the reverberation. Mean squared error effectively measures how far the estimated parameters are from the parameters chosen by the user. This means that while the classification accuracy may be high, so the effect and the users agree most of the time, the differences on the parts they do not agree may be too high for the model to be useful. Therefore, the most useful model is the model with the least mean squared error. In our case, the multi-class SVM approach performs best regarding MSE in all but one case, while it performs similar to the GNB in regards to f1-scores.

Table 5. Mean Squared Errors for the normalized parameters. Lowest MSEs for each sets are in bold. $|\mathcal{C}|$ is the number of classes calculated for each set.

| Training Set | $\|\mathcal{C}\|$ | GNB | SVM | HMM | $\text{HMM}^{\text{SVM}}$ |
|---|---|---|---|---|---|
| 1 | 7 | 0.0067 | **0.0065** | 0.0114 | 0.0117 |
| 2 | 9 | 0.0015 | **0.0010** | 0.0025 | 0.0045 |
| 3 | 6 | **0.0091** | 0.0097 | 0.0106 | 0.0096 |
| 4 | 8 | **0.0014** | **0.0014** | 0.0035 | 0.0062 |
| 5 | 7 | 0.0082 | **0.0047** | 0.0069 | 0.0135 |
| 6 | 7 | 0.0044 | **0.0041** | 0.0066 | 0.0204 |

Table 6. Weighted f1-scores for the user-trained models. Highest scores for each user are in bold. $|\mathcal{C}|$ is the number of classes calculated for each user.

| User | $\|\mathcal{C}\|$ | GNB | SVM | HMM | $\text{HMM}^{\text{SVM}}$ |
|---|---|---|---|---|---|
| A | 30 | **0.79** | 0.73 | 0.06 | 0.11 |
| B | 22 | **0.74** | 0.66 | 0.17 | 0.16 |
| C | 32 | 0.81 | **0.84** | 0.12 | 0.18 |

Table 7. MSEs for the user-trained models. Lowest MSEs for each user are in bold. $|C|$ is the number of classes calculated for each user.

| User | $\|\mathcal{C}\|$ | GNB | SVM | HMM | $\text{HMM}^{\text{SVM}}$ |
|---|---|---|---|---|---|
| A | 30 | **0.0104** | 0.0138 | 0.0510 | 0.0568 |
| B | 22 | **0.0141** | 0.0226 | 0.0538 | 0.0386 |
| C | 32 | **0.0087** | 0.0091 | 0.0444 | 0.0480 |

### 6.2 Perceptual Evaluation

Perceptual evaluation of the data was performed using multi-stimulus MUSHRA-style listening tests in the WEB AUDIO EVALUATION TOOL (WAET) [26]. This was in order to check how our models performed when trained by different users of the reverberation effect.

For this test we used 33 audio files from our dataset. We normalized them in regards to mean loudness and converted them to mono. We used 3 expert users of the reverberation effect from the Centre of Digital Music to train our system by applying suitable reverberation to each of them. For each of the "trainers," we kept the parameters they used for 27 of those files and trained our models as described in Sec. 5 (Classification performance for each of those models can be seen in Tables 6 and 7). Using the GNB and SVM models for each trainer, we then applied automatic reverberation to the 6 remaining files. These files consisted of excerpts from 2 singing tracks, a bass guitar, a saxophone, a drum, and a piano track.

For each file, we created a multi-stimulus trial. Each of the trials included a visible outer reference (the original file with reverberation applied manually by one of the three

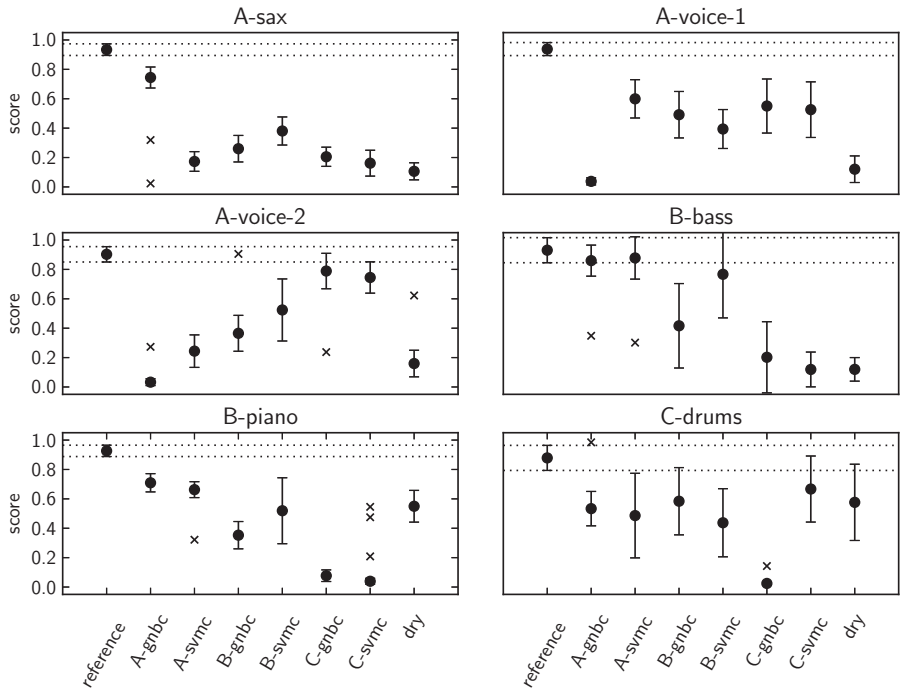


Fig. 4. Results of the MUSHRA-style tests. The bars represent the upper and lower limits for the 95% confidence intervals. Full circles are mean values, x symbol points are outliers, and dotted lines represent the upper and lower standard error borders of the reference. On the x-axis are the labels of the stimuli. Each of the letters A, B, or C represents a reverberated track generated from a model trained by the corresponding expert. Suffixes -svmc and -gnbc represent whether it was based on a Support Vector Machine or a Gaussian Naive Bayes classifier.

"trainers"), the same reference hidden in the stimuli, and an anchor (the original file with no reverberation applied to it). It also included six files with automatically applied reverberation (one from a GNB model and one from an SVM model, for each of the three "trainers" that trained those models). Subjects were asked to rate each of the stimuli in regards to how close it sounds to the reference.

Sixteen test subjects participated in the listening test. Those did not include the "trainers." They were mostly Ph.D. students and Post-doctoral associates from the Centre for Digital Music at Queen Mary University of London, with the exception of one student not from the Centre and a freelance employee. Three listeners were active users of the reverberation effect (two of them professionally), while the rest just knew what the effect sounded like. The average time of the test taken was 30 minutes and the test was considered difficult by most participants. The tests were all done using WAET in local mode on the desktop computer of the Media and Arts Technology studio control room at the same university.

Fig. 4 shows the mean rating, averaged over all participants, and the 95% intervals, for each stimulus in each trial. If our reverberator was successful, we would expect each model to be rated close to its respective reference, e.g., A-gnbc or A-svmc should be close to the reference for A-sax. For C-drums, A-sax, and A-voice-1 we can see that C-smvc, A-gnbc, and A-svmc score higher than the rest. For the case of B-bass we see that while the B- models were not rated closer than the rest, B-svmc is still very close to the reference. For B-piano, the models seem to perform poorly, while for A-voice-2, the A- models seem to have failed. In general for this small listening test, the tracks based on -svmc models appear more similar to the respective tracks with reverberation applied by the trainers. The listening test however fails to give very clear results. We suspect this was due to the difficulty of the question and the different concept of similarity for each subject.

## 7 CONCLUSION

From Tables 4–7 we can see that for our datasets, the non-sequential models performed better than the sequential counterparts, which performed comparably or even worse than the Naive Bayes classifier. This suggests that the Hidden Markov Models failed to capture correctly the temporal progression of our data. One of the reasons for this could be the onset segmentation method we use prior to feature extraction, which leads to uncorrelated feature vectors, as opposed to classical frame segmentation and thus damaging the Markov assumption. The disparity between the sequential and non-sequential classification results in Sec. 6.2 can also be attributed to the large number of classes that were produced as a result of the training by the users (and as a result, the smaller number of training samples for each class). The above suggest further exploration with different models and configurations. The best model so far seemed to be the One-vs-All Support Vector Machine classifier that performed best regarding weighted f1-score and Mean Squared Error. Our choice of models becomes clearer when

we take into account that our simple non-sequential models do not require past samples in order to make a decision, so we can use our models in real time with a minimum latency of 23 ms (a simple frame). This paper, in general, described an approach on a reverberation effect that could control a reverberator given desired characteristics of the impulse response and also remember those characteristics in the future. A possible implementation for such a system would be an audio effect that allows the user to select desired reverberation characteristics to be applied to specific tracks, and have the system suggest similar reverberation for newly introduced, but similar tracks.

## 8 LIMITATIONS/FUTURE WORK

While this initial approach appears promising, there are things to be desired regarding individual steps. Mathematically deriving characteristics of the reverberation does not necessarily lead to perceptually correct parameters. For example, impulses that are very closely placed together may not be perceived as distinct echoes, but Eq. (4) will count them as such. Figuring out more perceptually robust reverberation features will greatly improve this work. Another issue is that we did not take into account features relating to stereo signals such as Interaural Cross Correlation, Lateral Energy Fraction, Apparent Source Width, etc. Future research could take the direction of providing mappings between such features and low level parameters of a stereo reverberator. The architecture of the reverberator itself can be of concern. Moorer reverberators, although serve as a very good basis for our work given their simple design, are limited (for example they do not allow for independent control of early and late reverberation). One could try to exchange the current architecture with a more recent reverberator design [27] or even try to implement a model agnostic architecture so that it could be used with commercially available reverberation effects. ADEPT [28] provides a framework that could aid in the design of such a system. Regarding perceptual evaluation, there is work to be done on how to efficiently evaluate such systems. Our question on how "similar" the tracks with automatically applied reverb sounded to the reference, was deemed very difficult to answer by our test subjects, which made drawing conclusions difficult. One should use a more clearly defined objective for testing (e.g., reducing masking in a multi-track context).

The original Adaptive Digital Audio Effect architecture [13] supports multitrack DAFx, while our method has only been tested for effects applied on a single track. A logical next step would then be to extend our architecture to multitrack audio content. Also, our method is limited to pretrained sets of parameters, which can be limiting in the introduction of new unexplored audio. [29] describes a way to control continuous control parameters using discrete states using only two parameter states, something which would fit naturally with our approach. Another idea worth exploring is the combination of the approach described here with control using descriptive terms [10], [11]. Finally, the ability of our effect to be trained directly from measurements of reverberation, could possibly allow it to be trained directly from impulse responses or even reverberant sound samples. There are numerous works in the literature that would allow us to estimate reverberation time [30–33], echo density [34], clarity/definition [32], central time, and spectral centroid. [35] also gives an easily measurable set of features that correlate to subjective reverberation and which could be included with small alterations to our model.

## 9 ACKNOWLEDGMENT

This research has been sponsored by RPPtv Ltd.

## 10 REFERENCES

[1] Z. Rafii and B. Pardo "Learning to Control a Reverberator Using Subjective Perceptual Descriptors," *Int. Soc. Mus. Inf. Retr. (ISMIR)* (2009).

[2] J. A. Moorer "About This Reverberation Business," *Comput. Music J.*, vol. 3, no. 2, pp. 13–28 (1979 Jun.).

[3] ITUR Recommendation, "Bs. 1534-1. Method for the Subjective Assessment of Intermediate Sound Quality (MUSHRA)," *Intl. Tel. Union, Geneva* (2001).

[4] E. T. Chourdakis and J. D. Reiss "Automatic Control of a Digital Reverberation Effect Using Hybrid Models," presented at the *AES 60th International Conference: DREAMS (Dereverberation and Reverberation of Audio, Music, and Speech)* (2016 Jan.), conference paper 9-2.

[5] J. Scott et al., "Automatic Multi-Track Mixing Using Linear Dynamical Systems," *Proceedings of Sound Music Computing (SMC) Conference* (2011).

[6] S. Hafezi and J. D. Reiss "Autonomous Multitrack Equalization Based on Masking Reduction," *J. Audio Eng. Soc.*, vol. 63, pp. 312–323 (2015 May).

[7] E. P. Gonzalez and J. D. Reiss "A Real-Time Semiautonomous Audio Panning System for Music Mixing," *J. Adv. Sign. Proc.*, vol. 2010, no. 1, pp. 1–10 (2010).

[8] Z. Ma et al., "Intelligent Dynamic Range Compression," *J. Audio Eng. Soc.*, vol. 63, pp. 412–426 (2015 Jun.).

[9] T. Carpentier, M. Noisternig, and O. Warusfel "Hybrid Reverberation Processor with Perceptual Control," *17th Int. Conf. on Digital Audio Effects* (2014).

[10] R. Stables et al., "SAFE: A System for the Extraction and Retrieval of Semantic Audio Descriptors," *Int. Soc. Mus. Inf. Retr. (ISMIR)* (2014).

[11] P. Seetharaman and B. Pardo "Crowdsourcing a Reverberation Descriptor Map," *22nd ACM Int. Conf. on Multimedia.* (2014).

[12] J. Scott and Y. E. Kim "Instrument Identification Informed Multi-Track Mixing," *Int. Soc. Mus. Inf. Retr. (ISMIR)* (2013).

[13] V. Verfaille, U. Zölzer, and D. Arib "Adaptive Digital Audio Effects (A-DAFx): A New Class of Sound Transformations," *IEEE Trans. Audio, Speech, Language Process.*, vol. 14, pp. 1817–1831 (2006).

[14] Z. Rafii and B. Pardo "A Digital Reverberator Controlled through Measures of the Reverberation," Technical Report, Northwestern University (2009).

[15] G. Peeters "A Large Set of Audio Features for Sound Description (Similarity and Classification) in the

Cuidado Project," Technical Report, IRCAM (2004).

[16] T. Ganchev, N. Fakotakis, and G. Kokkinakis "Comparative Evaluation of Various MFCC Implementations on the Speaker Verification Task," *10th Int. Conf. Speech. Comp.* (2005).

[17] J. Foote "Automatic Audio Segmentation Using a Measure of Audio Novelty," *IEEE Int. Conf. Multimed. Expo.*, vol. 1, pp. 42–455 (2000).

[18] K. West and S. Cox "Finding an Optimal Segmentation for Audio Genre Classification," *Int. Soc. Mus. Inf. Retr. (ISMIR)* (2005).

[19] Y. Lu et al., "Feature Selection Using Principal Feature Analysis," *15th ACM Int. Conf. on Multim.*, vol. 1, pp. 27–30 (IEEE, 2007).

[20] C. M. Bishop *Pattern Recognition and Machine Learning* (Springer, 2006).

[21] M. Aly "Survey on Multiclass Classification Methods," *Neural Networks*, pp. 1–9 (2005).

[22] J. Platt, "Probabilistic Outputs for Support Vector Machines and Comparison to Regularize Likelihood Methods," *Advances in Large Margin Classifiers*, pp. 61–74 (MIT Press, 2000).

[23] B. De Man, M. Mora-Mcginity, G. Fazekas, and J. D. Reiss "The Open Multitrack Testbed," presented at the *137th Convention of the Audio Engineering Society* (2014 Oct.), eBrief 165.

[24] E. Jones et al., "SciPy: Open Source Scientific Tools for Python, 2001—," [Online; accessed 2016-03-29].

[25] D. Bogdanov et al., "Essentia: An Audio Analysis Library for Music Information Retrieval," *Int. Soc. Mus. Inf. Retr. (ISMIR)* (2013).

[26] N. Jillings et al., "Web Audio Evaluation Tool: A Framework for Subjective Assessment of Audio," *2nd Web Audio Conf.* (April 2016).

[27] V. Välimäki, J. Parker, L. Savioja, J. O. Smith, and J. Abel "More than 50 Years of Artificial Reverberation," presented at the *AES 60th International Conference: DREAMS (Dereverberation and Reverberation of Audio, Music, and Speech)* (2016 Jan.), conference paper K-1.

[28] O. Campbell, C. Roads, A. Cabrera, M. Wright, and Y. Visell "ADEPT: A Framework for Adaptive Digital Audio Effects," *2nd AES Workshop on Intel. Mus. Prod.* (2016 Sep.).

[29] J. Pazis and M. G. Lagoudakis "Binary Action Search for Learning Continuous-Action Control Policies," *Int. Conf. Mach. Learn. (ICML)* (2009).

[30] T. J. Cox, F. Li, and P. Darlington "Extracting Room Reverberation Time from Speech Using Artificial Neural Networks," *J. Audio Eng. Soc.*, vol. 49, pp. 219–230 (2001 Apr.).

[31] J. F. Santos and T. H. Falk "Blind Room Acoustics Characterization Using Recurrent Neural Networks and Modulation Spectrum Dynamics," presented at the *AES 60th International Conference: DREAMS (Dereverberation and Reverberation of Audio, Music, and Speech)* (2016 Jan.), conference paper 3-1.

[32] A. Belhomme et al., "Blind Estimation of Room Acoustic Parameters Using Kernel Regression," presented at the *AES 60th International Conference: DREAMS (Dereverberation and Reverberation of Audio, Music, and Speech)* (2016 Jan.), conference paper 1-1.

[33] I. J. Kelly, F. M. Boland, and J. Skoglund "Robust Estimation of Reverberation Time Using Polynomial Roots," presented at the *AES 60th International Conference: DREAMS (Dereverberation and Reverberation of Audio, Music, and Speech)* (2016 Jan.), conference paper 6-1.

[34] J. S. Abel and P. Huang "A Simple, Robust Measure of Reverberation Echo Density," presented at the *121st Convention of the Audio Engineering Society* (2006 Oct.), convention paper 6985.

[35] E. Kahle and J.-P. Jullien "Some New Considerations on the Subjective Impression of Reverberance and its Correlation with Objective Criteria," *127th Annual Meeting of Acoustic Soc. of America* (1994).

[36] S. E. Krüger et al., "Speech Recognition with Support Vector Machines in a Hybrid System," *INTERSPEECH 2005 - Eurospeech, 9th European Conference on Speech Communication and Technology, Lisbon, Portugal, September 4–8* (2005).

## APPENDIX

### A.1 Hidden Markov Models with SVM Emissions

The Gaussian HMM classifiers described above work sufficiently, but they rely on Gaussian probability density functions (PDF) for their emissions which cannot discriminate well. On the other hand, SVMs discriminate well even with very few samples but do not provide emission PDFs.We would like to combine the discriminating power of the SVMs with the sequential nature of the HMMs. We can use some tricks to derive a PDF.

Using Platt's method [22] and following the work done in [36] for our SVM Classifier we have for a class $k$ and a sample feature vector $\boldsymbol{\phi}_i$ [36]:

$$p(C_k|\boldsymbol{\phi}_i) = 1 \Big/ \left[ \sum_{l=1, l\neq k}^{K} \frac{1}{\mu_{kl}} - (K-2) \right] \tag{13}$$

where $\mu_{kl}$ is the probability that the class is either $k$ or $l$, that is:

$$\mu_{kl} = p(C_k \text{ or } C_l|\boldsymbol{\phi}_i) \tag{14}$$

If we regard the class of $f_i$ our HMM's state, then we can compute its emission probabilities by using the Bayes rule:

$$p(\boldsymbol{\phi}_i|C_k) = \alpha \frac{p(C_k|\boldsymbol{\phi}_i)}{p(C_k)} \tag{15}$$

where $p(C_k)$ can be estimated by counting occurrences of $C_k$ in our data and $\alpha$ is the normalization factor. Since our model's states are the same as our classes, the process of classification reduces to predicting the hidden sequence state of our HMM using the Viterbi algorithm.

Training of the HMM in that case is achieved by first training the SVMs that will provide the emission probabilities, then normally training the HMM using the VITERBI or the BAUM-WELCH (FORWARD-BACKWARD) [20] algorithm.

### A.2 Mapping from Desired IR Characteristics to Filter Parameters

In Sec. 3 we present the mapping that allows us to approximate measurements of reverberation ($T_{60}$, $D$, $T_c$, $C$, $S_c$) given the filter parameters ($g_{1..6}$, $d_{1..6}$, $d_a$, $G$, $g_c$). In order to control our effect directly we need to calculate filter parameters from such approximations. In the case of $S_c$ this is easy since it is just dependent on $g_c$, but the rest give us a non-convex $4 \times 4$ system. Given that the filter parameters constitute our variables, and their values are also constrained (between 0 and 1 for the gains) finding the feasible space of that system is non-trivial. In some cases it is impossible to have an exact solution (for example it is impossible to have an arbitrarily low echo density and a very low reverb time). Instead, we approximate a solution that minimizes an objection function that brings us to a non-exact, but hopefully good enough choice of parameters.

Suppose we have a vector of desired reverberation characteristics (remember we can directly estimate $g_c$ from $S_c$):

$$\boldsymbol{v} = \begin{bmatrix} T_{60} & D & C & T_c \end{bmatrix}^T \tag{16}$$

and the desired reverberation characteristics:

$$\boldsymbol{v'} = \begin{bmatrix} T'_{60} & D' & C' & T'_c \end{bmatrix}^T \tag{17}$$

we need to find a set of parameters that minimizes the Euclidean distance of the target characteristics from the actual measurements, given the constraints of our parameters. Furthermore, we add the extra constraint of a uniform error distribution.

We find the optimal solution for the problem below (all variables are normalized to 0–1):

$$\begin{aligned} &\underset{\boldsymbol{x}=[g_1\ d_1\ d_a\ G]^T}{\text{minimize:}}\ f_0(\boldsymbol{x}) = \sqrt{\boldsymbol{e}^T\boldsymbol{e}} + \text{Var}[\boldsymbol{e}]^2 \\ &\text{subject to:} \\ &\quad 0 < g_1 < 1, \\ &\quad 0 < G < 1, \\ &\quad d_{1,min} \le d_1 \le d_{1,max}, \\ &\quad d_{a,min} \le d_a \le d_{a,max} \\ &\text{where:} \\ &\quad \boldsymbol{e} = \boldsymbol{v} - \boldsymbol{v'} \end{aligned} \tag{18}$$

Analytically the $4 \times 4$ system of equations leads to: (Note that $g_c$ and $g_a$ are treated as constants.)

(1) From $C'$ and $g_1$ we can derive $G$:

$$G = f_1(C', g_1) = A(g_1) \cdot 10^{-\frac{C'}{20}} \tag{19}$$

(2) From $C'$, $T'_{60}$ and $g_1$ we can derive $d_1$:

$$\begin{aligned} d_1 &= f_2(C', T'_{60}, g_1) \\ &= \frac{T'_{60}\log(g_1)}{0.05 \cdot C \cdot \log(10) - \log(A(g_1)) - 3 \cdot \log(10)} \end{aligned} \tag{20}$$

(3) from $D'$ and $d_1$, we can derive $d_a$:

$$d_a = \frac{2.078125}{d_1 \cdot D'} \tag{21}$$

where:

$$A(g_1) = \sqrt{\frac{2(1+g_c)}{(1-g_c)\sum_{k=1}^{6}\frac{g_1^{2\cdot 1.5^{-k+1}}}{1-g_1^{2\cdot 1.5^{-k+1}}}}} \tag{22}$$

So if we could pick the correct value of $g_1$ and we have the target values $T'_{60}$, $C'$, $D'$, and $T'_c$ we can derive the other three parameters. Unfortunately, it is non-trivial to find a closed form solution but we find a value for $g_1$ numerically given our constraints.

We can rewrite the optimization problem above as:

$$\begin{aligned} &\underset{g_1}{\text{minimize:}}\ f_0(g_1) = \sqrt{\boldsymbol{e}^T\boldsymbol{e}} + \text{Var}[\boldsymbol{e}]^2 \\ &\text{subject to:} \\ &\quad 0 < g_1 < 1, \\ &\text{where:} \\ &\quad G = f_1(C', g_1) \\ &\quad d_1 = f_2(C', T'_{60}, g_1) \\ &\quad d_a = f_3(C', T'_{60}, D', g_1) \end{aligned} \tag{23}$$

We managed to reduce the original $4 \times 4$ optimization problem to a numerical analysis problem of just one variable (which can be easily be solved by the line-search solver of our choice), the solution to which gives us a sub-optimal solution when there is no exact solution in our feasible space, and the exact solution when there is one. If we derive those measurements directly from an impulse response, we expect the problem to have an exact solution. We only expect non-exact solutions when the reverberation measurements are chosen arbitrarily.

## THE AUTHORS

Emmanouil Chourdakis

Emmanouil Chourdakis is a Ph.D. student at the Centre for Digital Music at Queen Mary University of London. He has received an M.Sc. in digital music processing from the Queen Mary University of London in 2013 and an electronic and computer engineering diploma from the Technical University of Crete in 2011. During his studies he has worked on a variety of subjects including computer aided composition systems, real-time DAFx, and machine learning applied to DAFx with most notable his Master Thesis on intelligent reverberation, which was rewarded with the Michael Clark Prize for best Electronic Engineering Project.

Josh Reiss

Josh Reiss is a Reader with Queen Mary University of London's Centre for Digital Music, where he leads the audio engineering research team. Dr. Reiss has published over 160 scientific papers. His co-authored publications, "Loudness Measurement of Multitrack Audio Content Using Modifications of ITU-R BS.1770" and "Physically Derived Synthesis Model of an Aeolian Tone," were recipients of the 134th AES Convention's Best Peer Reviewed Paper Award and the 141st AES Convention's Best Student Paper Award, respectively. He co-authored the textbook *Audio Effects: Theory, Implementation and Application*. He is a former governor of the AES and was General Chair of the 128th, Program Chair of the 130th and co-Program Chair of the 138th AES Conventions.